Proprietary Flex SDK 3.1.0.0 GA
Gecko SDK Suite 3.1
December 9, 2020

The Proprietary Flex SDK is a complete software development suite for proprietary wireless applications. Per its namesake, Flex offers two implementation options.

The first uses Silicon Labs RAIL (Radio Abstraction Interface Layer), an intuitive and easily-customizable radio interface layer designed to support both proprietary and standards-based wireless protocols.

The second uses Silicon Labs Connect, an IEEE 802.15.4-based networking stack designed for customizable broad-based proprietary wireless networking solutions that require low power consumption and operates in either the sub-GHz or 2.4 GHz frequency bands. The solution is targeted towards simple network topologies.

The Flex SDK is supplied with extensive documentation and sample applications. All examples are provided in source code within the Flex SDK sample applications.

These release notes cover SDK version(s):

3.1.0.0 released December 9, 2020

Proprietary

CONNECT APPS AND STACK KEY FEATURES

- EFR32xG21 and MGMx CONNECT GA support
- FreeRTOS Support for CONNECT
- Long Range PHY Support for CONNECT
- New example: ECDH key exchange
- Example ported from Flex 2.7: DMP Light/Switch
- Support for the RAIL-embedded IEEE 802.15.4 PHY

RAIL APPS AND LIBRARY KEY FEATURES

- EFR32FG23 Alpha 2 Support
- EFR32xG21 Flex Support
- Wi-SUN Profile and PHYs in Radio Configurator
- New example: SimpleTRX Standards for demonstrating 15.4/BLE usage

Compatibility and Use Notices

If you are new to the Silicon Labs Flex SDK, see Using This Release.

Compatible Compilers:

IAR Embedded Workbench for ARM (IAR-EWARM) version 8.30.1

- Using wine to build with the IarBuild.exe command line utility or IAR Embedded Workbench GUI on macOS or Linux could result in incorrect files being used due to collisions in wine's hashing algorithm for generating short file names.
- Customers on macOS or Linux are advised not to build with IAR outside of Simplicity Studio. Customers who do should carefully verify that the correct files are being used.

GCC (The GNU Compiler Collection) version 7.2.1, provided with Simplicity Studio.

Contents

1 Connect Applications 1
1.1 New Items 1
1.2 Improvements 1
1.3 Fixed Issues 1
1.4 Known Issues in the Current Release 1
1.5 Deprecated Items 1
1.6 Removed Items 1
2 Connect Stack 2
2.1 New Items 2
2.2 Improvements 2
2.3 Fixed Issues 2
2.4 Known Issues in the Current Release 2
2.5 Deprecated Items 2
2.6 Removed Items 2
3 RAIL Applications 3
3.1 New Items 3
3.2 Improvements 3
3.3 Fixed Issues 3
3.4 Known Issues in the Current Release 3
3.5 Deprecated Items 3
3.6 Removed Items 3
4 RAIL Library 4
4.1 New Items 4
4.2 Improvements 4
4.3 Fixed Issues 4
4.4 Known Issues in the Current Release 5
4.5 Deprecated Items 5
4.6 Removed Items 5
5 Using This Release 6
5.1 Installation and Use 6
5.2 Support 6
6 Legal 7
6.1 Disclaimer 7
6.2 Trademark Information 7

1 Connect Applications

1.1 New Items

Added in release 3.1.0.0

- New Applications
  - Flex (Connect) - SoC ECDH Key Exchange
  - Flex (Connect) - SoC Light Example DMP
  - Flex (Connect) - SoC Switch Example
- Support for EFR32xG21 via Std IEEE 802.15.4 PHY – Each Application
- Support for MGMx modules via Std IEEE 802.15.4 PHY – Each Application
- FreeRTOS (besides baremetal and Micrium) support – Each Application
- Long Range PHY support – Each Application

1.2 Improvements

Changed in release 3.1.0.0

- Application code structure is made simpler, `app_callbacks.c` is removed from each Application:
  - `emberAfInitCallback()` is moved to `app_init.c`
  - the other application related callbacks are moved to `app_process.c`
- GBL generation support for each Application via script mechanism
  - `connect_create_gbl_image.bat`
  - `connect_create_gbl_image.sh`
- Remove unnecessary Zigbee-derived housekeeping at the following Applications:
  - Flex (Connect) - SoC Sensor
  - Flex (Connect) - SoC Sink
- Radio Configurator UI improvement: only the Application relevant Profiles are selectable

1.3 Fixed Issues

None

1.4 Known Issues in the Current Release

Issues in bold were added since the previous release. If you have missed a release, recent release notes are available on https://www.si-labs.com/products/software.

ID #	Description	Workaround
	Connect NCP-Host applications are not supported.	

1.5 Deprecated Items

None

1.6 Removed Items

None

2 Connect Stack

2.1 New Items

Added in release 3.1.0.0

- Added support for EFR32xG21.
- Added support for MGMx modules.
- Added a new Radio Stream plugin that exposes APIs `emberStartTxStream()` and `emberStopTxStream()` to start and stop radio streams. PN9 and CW modes are supported
- Added a new API `emberApplyIrCalibration()` that applies a previous calibration value.
- Added a new API `emberTempCalibration()` that performs a temperature calibration.
- Added a new API `emberGetCalType()` that indicates which calibration is needed. Should be called whenever `emberAfRadioNeedsCalibratingCallback()` is fired prior to calling any calibration function
- Added a new API `emberGetChildInfo()` to ask a coordinator if an address exists in the network
- Added a new API `emberSetRadioChannelExtended()` that allows to set a channel without a flash write

2.2 Improvements

Changed in release 3.1.0.0

- Connect Stack IPC plugin now relies on CMSIS-RTOS API v2. Micrium Stack IPC plugin was renamed CMSIS Stack IPC. Stack tasks priorities can now be configured from the Project Configuration UI.
- `emberInit()` is no longer calibrating the radio. The image rejection calibration can now be achieved by calling `emberCalibrateCurrentChannel()` whenever `emberAfRadioNeedsCalibratingCallback()` is fired. **Note**: The default callback makes a call to `emberCalibrateCurrentChannel()`.
- Improve API error values. A plugin API now returns EMBER_LIBRARY_NOT_PRESENT if the plugin is not installed.

2.3 Fixed Issues

None

2.4 Known Issues in the Current Release

Issues in bold were added since the previous release. If you have missed a release, recent release notes are available on https://www.silabs.com/products/software.

ID #	Description	Workaround
501561	In the Legacy HAL component, the PA configuration is hard-coded regardless of the user or board settings.	Until this is changed to properly pull from the configuration header, the file ember-phy.c in the user's project will need to be modified by hand to reflect the desired PA mode, voltage, and ramp time.

2.5 Deprecated Items

None

2.6 Removed Items

None

3 RAIL Applications

3.1 New Items

Added in release 3.1.0.0

- New Applications:
    - Flex (RAIL) - Simple TRX Standards
- Support for EFR32FG23
    - Flex (RAIL) - Burst Duty Cycle
    - Flex (RAIL) - Energy Mode
    - Flex (RAIL) - Light
    - Flex (RAIL) - Long Preamble Duty Cycle
    - Flex (RAIL) - Empty Example
    - Flex (RAIL) - Range Test
    - Flex (RAIL) - Simple TRX
    - Flex (RAIL) - Simple TRX with Auto-ACK
    - Flex (RAIL) - Simple TRX Standards
    - Flex (RAIL) - Switch
    - Flex (RAIL) - Wireless M-bus Collector
    - Flex (RAIL) - Wireless M-bus Meter
- Wi-Sun Profile support at Flex (RAIL) - Simple TRX
- Support for EFR32xG21 via Std IEEE 802.15.4 PHY
    - Flex (RAIL) - Simple TRX Standards

3.2 Improvements

Changed in release 3.1.0.0

- Radio Configurator UI improvement: only the Application relevant Profiles are selectable

3.3 Fixed Issues

None

3.4 Known Issues in the Current Release

None

3.5 Deprecated Items

None

3.6 Removed Items

None

4 RAIL Library

4.1 New Items

Added in release 3.1.0.0

- Added new API RAIL_GetRadioStateDetail() that provides more detailed radio state information than RAIL_GetRadioState.
- Added RAIL_RxPacketInfo_t::filterMask field of type RAIL_AddrFilterMask_t, which is a bitmask representing which address filter(s) the packet has passed.
- Added the ability for RAIL_GetRssi() to wait for a valid RSSI in radio states that are transitioning into RX. Additionally, a maximum wait timeout for a valid RSSI can be configured using the new API RAIL_GetRssiAlt().
- Added a new RAIL_ZWAVE_ConfigRxChannelHopping() API to configure Z-Wave Rx channel hopping using the recommended hopping parameters.
- Added a new RAIL_ZWAVE_GetRegion() API to determine the currently selected Z-Wave region.
- Added a new RAIL_SupportsTxPowerModeAlt() API to get the minimum and maximum power levels for a specific power mode if the power mode is supported by the chip.
- Added a new API RAIL_SetAddressFilterAddressMask() that allows for setting a bit mask pattern for packet data in the address filters.
- Added support for MGM210PB22JIA, MGM210PB32JIA, BGM210PB22JIA and BGM210PA32JIA modules.
- Added an event RAIL_EVENT_PA_PROTECTION to indicate the power protection circuit has kicked in.
- Created a “RAIL Utility, Callbacks” component for application-level callbacks.

4.2 Improvements

Changed in release 3.1.0.0

- Added support for reporting more detailed transmit errors on the Packet Trace Interface (PTI).
- Updated RAIL_ZWAVE_ReceiveBeam() to automatically idle the radio when RAIL_ZWAVE_ReceiveBeam() finishes even when no beam is detected.
- Added the ability to use the “RAIL Utility, Initialization” component multiple times when creating a multiprotocol application.
- Changed RAIL_PacketTimeStamp_t::totalPacketBytes from uint32_t to uint16_t to reduce RAM usage.
- The “RAIL Utility, Initialization” component now defaults most options to a disabled state, instead of enabled. Now you have to opt-in, instead of opt-out, of RAIL init functionality.
- The “RAIL Utility, PA” component now enables PA calibration by default to ensure that PA power remains consistent chip-to-chip.
- Add new RAIL_EVENT_RF_SENSED as an alternative to the current RAIL_StartRfSense() callback parameter.
- Added a new API RAIL_ConfigSleepAlt() to allow configuring the PRS channel, RTCC channel, and whether sleep is enabled in one call.
- Created a new “RAIL Utility, Protocol” component for setting up RAIL to use one of the standards based PHYs by default.
- In multiprotocol RAIL, when the supplied handle is not the active handle, RAIL_GetRadioState now returns RAIL_RF_STATE_RX rather than RAIL_RF_STATE_IDLE if a background receive is currently scheduled.
- Antenna diversity settings for xGM210 modules are now split in a new config file: sl_rail_util_ant_div_config.h.

4.3 Fixed Issues

Fixed in release 3.1.0.0

ID #	Description
362133	The default RSSI offset on the EFR32xG1, EFR32xG12, EFR32xG13, EFR32xG14, and EFR32xG21 chips does not compensate for a known internal hardware offset. This offset is chip specific and can be found using the new “RAIL Utility, RSSI” component which will load the correct value for your chip by default when the plugin is enabled. Since the hardware and antenna design can also impact this offset it is recommended that you measure this value for your particular hardware for the best accuracy. This correction is not enabled by default on the chips listed above to prevent changing radio behavior significantly without the user opting into this change. For the EFR32xG22 and future chips the hardware offset is measured and included by default.

ID #	Description
471715	Fixed an issue when using RAIL_ConfigAntenna() on the EFR32xG22 with an RF path other than 0 since these parts do not have multiple RF paths.
519195	The EFR32xG21 will now use RTCC channel 0, as opposed to the PRORTC, to perform sleep timer synchronization. This will help lower the EM2 current consumption for this chip.
630457	On custom boards, the “RAIL Utility, PTI“ component no longer reserves pins for use without being configured.
632723	The EFR32xG22 will limit going to EM1P sleep mode when an 80MHz HRFRCO PLL system clock is selected. Going to EM1P sleep is not supported when using the DPLL on this hardware as it can cause clock drift which would impact radio timing and tuning.
638067	Fixed a DMP issue that poached transmit power when switching between protocols using the same channel configuration and channel.
639833	Fixed a potential radio hang on a corrupted BLE packet when doing BLE AoX.
642893	Reduced RAIL library flash data alignment needs on the EFR32xG22.
645641	Fixed an EFR32xG22 issue where a state transition to receive after a transmit from EM2 sleep would drop packets.

4.4 Known Issues in the Current Release

None

4.5 Deprecated Items

None

4.6 Removed Items

None

5 Using This Release

This release contains the following

- Radio Abstraction Interface Layer (RAIL) stack library
- Connect Stack Library
- RAIL and Connect Sample Applications
- RAIL and Connect Components and Application Framework

This SDK depends on Gecko Platform. The Gecko Platform code provides functionality that supports protocol plugins and APIs in the form of drivers and other lower layer features that interact directly with Silicon Labs chips and modules. Gecko Platform components include EMLIB, EMDRV, RAIL Library, NVM3, and mbedTLS. Gecko Platform release notes are available through Simplicity Studio's Documentation tab.

For more information about the Flex SDK v3.x see UG103.13: RAIL Fundamentals and UG103.12: Silicon Labs Connect Fundamentals. If you are a first time user, see QSG168: Proprietary Flex SDK v3.x Quick Start Guide.

5.1 Installation and Use

Stack installation instruction are covered in the Simplicity Studio 5 online User's Guide.

Use the Flex SDK v3.x with the Silicon Labs Simplicity Studio 5 development platform. Simplicity Studio ensures that most software and tool compatibilities are managed correctly. Install software and board firmware updates promptly when you are notified.

Documentation specific to the SDK version is installed with the SDK. Additional information can often be found in the knowledge base articles (KBAs). API references and other information about this and earlier releases is available on https://docs.silabs.com/.

5.2 Support

Development Kit customers are eligible for training and technical support. Use the Silicon Labs Flex web page to obtain information about all Silicon Labs Thread products and services, and to sign up for product support.

You can contact Silicon Laboratories support at http://www.silabs.com/support.

6 Legal

6.1 Disclaimer

Silicon Labs intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Labs products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Labs reserves the right to make changes without further notice to the product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Without prior notification, Silicon Labs may update product firmware during the manufacturing process for security or reliability reasons. Such changes will not alter the specifications or the performance of the product. Silicon Labs shall have no liability for the consequences of use of the information supplied in this document. This document does not imply or expressly grant any license to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any FDA Class III devices, applications for which FDA premarket approval is required, or Life Support Systems without the specific written consent of Silicon Labs. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Labs products are not designed or authorized for military applications. Silicon Labs products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons. Silicon Labs disclaims all express and implied warranties and shall not be responsible or liable for any injuries or damages related to use of a Silicon Labs product in such unauthorized applications.

6.2 Trademark Information

Silicon Laboratories Inc.®, Silicon Laboratories®, Silicon Labs®, SiLabs® and the Silicon Labs logo®, Bluegiga®, Bluegiga Logo®, ClockBuilder®, CMEMS®, DSPLL®, EFM®, EFM32®, EFR, Ember®, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZRadio®, EZRadioPRO®, Gecko®, Gecko OS, Gecko OS Studio, ISOmodem®, Precision32®, ProSLIC®, Simplicity Studio®, SiPHY®, Telegesis, the Telegesis Logo®, USBXpress®, Zentri, the Zentri logo and Zentri DMS, Z-Wave®, and others are trademarks or registered trademarks of Silicon Labs. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. Wi-Fi is a registered trademark of the Wi-Fi Alliance. All other products or brand names mentioned herein are trademarks of their respective holders.